LIS User's Guide

Submitted under Task Agreement GSFC-CT-2

Cooperative Agreement Notice (CAN)
CAN-00OES-01

Increasing Interoperability and Performance of Grand Challenge Applications in the Earth, Space, Life, and Microgravity Sciences

May 28, 2003

Revision 1.1

History:

Revision	Summary of Changes	Date
1.1	Milestone F release	April 25, 2003
1.0	Milestone F submission	March 31, 2003

Contents

1 Introduction

This is a draft of LIS' User's Guide. This document describes how to download and install the code and data needed to run the LIS executable for LIS' "First Code Improvement" milestone – Milestone F. It describes how to build and run the code, and finally this document also describes how to download output data sets to use for validation. Updates to this document will provide more detailed instructions on how to configure the executable and will address the graphical user interface.

This document consists of 8 sections, described as followed:

1 **Introduction**: the section you are currently reading

2 **Background**: general info. about the LIS project

3 **Preliminaries**: general info., steps, instructions, and definitions used throughout the rest of this document

4 **Obtaining the Source Code and Datasets**: the steps needed to download and install the code and data

5 **Building the Executable**: the steps needed to build the LIS executable

6 **Running the Executable**: the steps needed to prepare and submit a run, also describes the various run-time configurations

7 **Output Data Processing**: the steps needed to post-process generated output for visualization

8 **Retrieving "F" Output Data**: the steps needed to download data generated by the LIS team during its testing runs.

2 Background

This section provides some general information about the LIS project and land surface modeling.

2.1 LIS

The Land Information System (LIS) will have the following components: (1) A high-resolution (1km) Land Data Assimilation System (LDAS), involving several independent community Land Surface Models (LSMs), land surface data assimilation technologies, and integrated database operations for observation and prediction data management; and (2) A web-based user interface based on the GRid Analysis and Display System (GrADS) and the Distributed Oceanographic Data System (DODS) for accessing data mining, numerical modeling, and visualization tools. The LIS will be available as a "production" system on a centralized server for large applications. By incorporating and promulgating the existing Assistance for Land_surface Modeling Activities (ALMA; http://www.lmd.jussieu.fr/ALMA/) and DODS standards for model coupling and data visualization, LIS will contribute to the definition of the land surface modeling and assimilation standards for the Earth System Modeling Framework (ESMF).

2.2 Land Surface Modeling and Data Assimilation

In general, land surface modeling seeks to predict the terrestrial water, energy and biogeochemical processes by solving the governing equations of the soil-vegetation-snowpack medium. Land surface data assimilation seeks to synthesize data and land surface models to improve our ability to predict and understand these processes. The ability to predict terrestrial water, energy and biogeochemical processes is critical for applications in weather and climate prediction, agricultural forecasting, water resources management, hazard mitigation and mobility assessment.

In order to predict water, energy and biogeochemical processes using (typically 1-D vertical) partial differential equations, land surface models require three types of inputs: 1) initial conditions, which describe the initial state of land surface; 2) boundary conditions, which describe both the upper (atmospheric) fluxes or states also known as "forcings" and the lower (soil) fluxes or states; and 3) parameters, which are a function of soil, vegetation, topography, etc., and are used to solve the governing equations.

2.3 Land Data Assimilation System (LDAS)

LDAS is a model control and input/output system (consisting of a number of subroutines, modules written in Fortran 90 source code) that drives multiple offline one dimensional land surface models (LSMs) using a vegetation defined "tile" or "patch" approach to simulate sub-grid scale variability. The one-dimensional LSMs such as CLM and NOAH, which are subroutines of LDAS, apply the governing equations of the physical processes of the soil-vegetation-snowpack medium. These land surface models aim to characterize the transfer of mass, energy, and momentum between a vegetated surface and the atmosphere.

LDAS makes use of various satellite and ground based observation systems within a land data assimilation framework to produce optimal output fields of land surface states and fluxes. The LSM predictions are greatly improved through the use of a data assimilation environment such as the one provided by LDAS. In addition to being forced with real time output from numerical prediction models and satellite and radar precipitation measurements, LDAS derives model parameters from existing topography, vegetation and soil coverages. The model results are aggregated to various temporal and spatial scales, e.g., 3 hourly, $1/4°$.

The execution of LDAS starts with reading in the user specifications. The user selects the model domain and spatial resolution, the duration and timestep of the run, the land surface model, the type of forcing from a list of model and observation based data sources, the number of "tiles" per grid square (described below), the soil parameterization scheme, reading and writing of restart files, output specifications, and the functioning of several other enhancements including elevation correction and data assimilation.

The system then reads the vegetation information and assigns subgrid tiles on which to run the one-dimensional simulations. LDAS runs its 1-D land models on vegetation-based "tiles" to simulate variability below the scale of the model grid squares. A tile is not tied to a specific location within the grid square. Each tile represents the area covered by a given vegetation type.

Memory is dynamically allocated to the global variables, many of which exist within Fortran 90 modules. The model parameters are read and computed next. The time loop begins and forcing data is read, time/space interpolation is computed and modified as necessary. Forcing data is used to specify boundary conditions to the land surface model. The LSMs in LDAS are driven by atmospheric forcing data such as precipitation, radiation, wind speed, temperature, humidity, etc., from various sources. LDAS applies spatial interpolation to convert forcing data to the appropriate resolution required by the model. Since the forcing data is read in at certain regular intervals, LDAS also temporally interpolates time average or instantaneous data to that needed by the model at the current timestep. The selected model is run for a vector of "tiles", intermediate information is stored in modular arrays, and output and restart files are written at the specified output interval.

2.4 Community Land Model (CLM)

CLM is a 1-D land surface model, written in Fortran 90, developed by a grassroots collaboration of scientists who have an interest in making a general land model available for public use. LDAS currently uses CLM version 1.0, formerly known as Common Land Model. CLM version 2.0 was released in May 2002 and will be implemented in LDAS in future. The source code for CLM 2.0 is freely available from the National Center for Atmospheric Research (NCAR) (`http://www.cgd.ucar.edu/tss/clm/`). The CLM is used as the land model for the community climate system model (CCSM) (`http://www.ccsm.ucar.edu/`) and the community atmosphere model (CAM) (`http://www.cgd.ucar.edu/cms/`). CLM is executed with all forcing, parameters, dimensioning, output routines, and coupling performed by an external driver of the user's design (in this case done by LDAS). CLM requires pre-processed data such as the land surface type, soil and vegetation parameters, model initialization, and atmospheric boundary conditions as input. The model applies finite-difference spatial discretization methods and a fully implicit time-integration scheme to numerically integrate the governing equations. The model subroutines apply the governing equations of the physical processes of the soil-vegetation-snowpack medium, including the surface energy balance equation, Richards' [4] equation for soil hydraulics, the diffusion equation for soil heat transfer, the energy-mass balance equation for the snowpack, and the Collatz et al. [2] formulation for the conductance of canopy transpiration.

2.5 The Community NOAH Land Surface Model

The community NOAH Land Surface Model is a stand-alone, uncoupled, 1-D column model freely available at the National Centers for Environmental Prediction (NCEP; `ftp://ftp.ncep.noaa.gov/pub/gcp/ldas/noahlsm/`). NOAH can be executed in either coupled or uncoupled mode. It has been coupled with the operational NCEP mesoscale Eta model [1] and its companion Eta Data Assimilation System (EDAS) [5], and the NCEP Global Forecast System (GFS) and its companion Global Data Assimilation System (GDAS). When NOAH is executed in uncoupled mode, near-surface atmospheric forcing data (e.g., precipitation, radiation, wind speed, temperature, humidity) is required as input. NOAH simulates soil moisture (both liquid and frozen), soil temperature, skin temperature, snowpack depth, snowpack water equivalent, canopy water content, and the energy flux and water flux terms of the surface energy balance and surface water balance. The model applies finite-difference spatial discretization methods and a Crank-Nicholson time-integration scheme to numerically integrate the governing equations of the physical processes of the soil vegetation-snowpack medium, including the surface energy balance equation, Richards' [4] equation for soil hydraulics, the diffusion equation for soil heat transfer, the energy-mass balance equation for the snowpack, and the Jarvis [3] equation for the conductance of canopy transpiration.

2.6 Variable Infiltration Capacity Model

Variable Infiltration Capacity (VIC) model is a macroscale hydrologic model, written in C, being developed at the University of Washington, and Princeton University. The VIC code repository along with the model description and source code documentation is available `http://www.hydro.washington.edu/Lettenmaier/Models/VIC/VIChome.html`. VIC is used in macroscopic land use models such as SEA - BASINS (`http://boto.ocean.washington.edu/seasia/intro.htm`). VIC is a semi-distributed, grid-based hydrological model, which parameterizes the dominant hydrometeorological processes taking place at the land surface - atmospheric interface. The execution of VIC model requires preprocessed data such as precipitation, temperature, meteorological forcing, soil and vegetation parameters, etc. as input. The model uses three soil layers and one vegetation layer with energy and moisture fluxes exchanged between the layers. The VIC model represents surface and subsurface hydrologic processes on a spatially distributed (grid cell) basis. Partitioning grid cell areas to different vegetation classes can approximate sub-grid scale variation in vegetation characteristics. VIC models the processes governing the flux and storage of water and heat in each cell-sized system of vegetation and soil structure. The water balance portion of VIC is based on three concepts:

1) Division of grid-cell into fraction sub-grid vegetation coverage.

2) The variable infiltration curve for rainfall/runoff partitioning at the land surface.

3) A baseflow/deep soil moisture curve for lateral baseflow.

Water balance calculations are preformed at three soil layers and within a vegetation canopy. An energy balance is calculated at the land surface. A full description of algorithms in VIC can be found in the references listed at the VIC website.

3 Preliminaries

This code has been compiled and run on both SGI IRIX64 6.5 systems and Linux PC (Intel/AMD based) systems. These instructions expect that you are using such a system. In particular you need

Software:

- SGI
  - MIPSpro version 7.3.1.1m
  - Message Passing Toolkit, mpt, version 1.5.3.0
  - GNU's make, gmake, version 3.77
- Linux
  - Absoft's Pro Fortran Software Developement Kit, version 8.0
  - GNU's C and C++ compilers, gcc and g++, version 2.96
  - MPICH [1], version 1.2.4
  - GNU's make, gmake, version 3.77

System Resources:

- 250MB to 32GB of memory
- 2.5GB to 46GB of hard disk space
  - 64MB for source
  - 1.5 GB to 25GB for input data
  - 425MB to 20GB for output data

You need to create a working directory on your system that has sufficient disk space to install and run in. Throughout the rest of this document this directory shall be refered to as *$WORKING*.

[1] These libraries must have been built using Absoft's Pro Fortran SDK

4 Obtaining the Source Code and Datasets

This section describes how to obtain the source code and datasets needed to run the LIS executable.

1. Download the *source.tar.gz*, *scripts.tar.gz*, and the *postproc.tar.gz* files from http://lis2.sci.gsfc.nasa.gov:9090/Fcode/ into your working directory, *$WORKING*.

2. Download the *input.tar.gz*, *geos_forcing.tar.gz*, and the *gdas_forcing.tar.gz* files from http://lis2.sci.gsfc.nasa.gov:9090/Fdata/INPUT/ into your working directory, *$WORKING*.

3. Unpack the files. Run (in the order listed):

```
% gzip -dc input.tar.gz | tar xf -
% gzip -dc geos_forcing.tar.gz | tar xf -
% gzip -dc gdas_forcing.tar.gz | tar xf -
% gzip -dc source.tar.gz | tar xf -
% gzip -dc scripts.tar.gz | tar xf -
% gzip -dc postproc.tar.gz | tar xf -
```

4.1 Source files

Unpacking the *source.tar.gz* file will create a *$WORKING/LDAS/src-par* subdirectory. The structure of *src-par* is as follows:

Directory Name	Synopsis
clm2	Top level clm2 land surface model sub-directory
clm2/biogeochem	
clm2/biogeophys	Biogeophysics routines (e.g., surface fluxes)
clm2/camclm_share	Code shared between the clm2 and cam (e.g., calendar information)
clm2/csm_share	Code shared by all the geophysical model components of the Community Climate System Model (CCSM). Currently contains code for CCSM message passing orbital calculations and system utilities
clm2/ecosysdyn	Ecosystem dynamics routines (e.g., leaf and stem area index)
clm2/main	Control (driver) routines
clm2/mksrfdata	Routines for generating surface datasets
clm2/riverroute	River routing (RTM) routines
clm2/utils	Independent utility routines
driver	LIS' LDAS driver routines
iplib	Interpolation routines
lib	Libraries needed for linking
make	Makefile and needed headers
noah	NOAH land surface model

Source code documentation may be found on LIS's web-site at http://lis.gsfc.nasa.gov/Documentation/MilestoneF/Documentation/lis/index.html.

4.2 Input data files

The *input.tar.gz* file contains various static and seasonally varying input data files. Unpacking the *input.tar.gz* file will create a *$WORKING/LDAS/input* sub-directory. The structure of *input* is as follows:

Directory Name	Synopsis
BCS	Soil and Vegetation Classifications
GVEG	Land/Sea mask
AVHRR_LAI	Leaf Area Index files

The *geos_forcing.tar.gz* file contains one-month's worth (01 June 2001 through 30 June 2001) of GEOS forcing data. Unpacking the *geos_forcing.tar.gz* file will create a *FORCING/GEOS* sub-directory in *$WORKING/LDAS/input*.

The *gdas_forcing.tar.gz* file contains one-month's worth (01 June 2001 through 30 June 2001) of GDAS forcing data. Unpacking the *gdas_forcing.tar.gz* file will create a *FORCING/GDAS* sub-directory in *$WORKING/LDAS/input*.

In order to run at 5 km resolution, you need to download the 5 km input datasets. Files ending in .bfsa, .bin, and .gz are binary files. Files ending in .txt are flat ASCII files.

See Section 3; you will need the maximum values of the system resources described there in order to run at this resolution.

1. Download

   alb_01_5KM.bfsa
   alb_02_5KM.bfsa
   alb_03_5KM.bfsa
   alb_04_5KM.bfsa
   gfrac_01_5KM.bfsa
   gfrac_02_5KM.bfsa
   gfrac_03_5KM.bfsa
   gfrac_04_5KM.bfsa
   gfrac_05_5KM.bfsa
   gfrac_06_5KM.bfsa
   maxsnalb_5KM.bfsa
   tbot-uncr_5KM.bfsa

   from http://lis2.sci.gsfc.nasa.gov:9090/Fdata/INPUT/BCS/NOAH/ into *$WORKING/LDAS/input/BCS/5km/NOAH*

2. Download

   clay60_5KM.bfsa
   por60_5KM.bfsa

sand60_5KM.bfsa
silt60_5KM.bfsa
sim60soil5KM.txt
soicol60_5KM.bfsa

from http://lis2.sci.gsfc.nasa.gov:9090/Fdata/INPUT/BCS/ into *$WORKING/LDAS/input/BCS/5km*

3. Download

   UMD_605KM.txt.gz
   UMD_60mask5KM.txt.gz

   from http://lis2.sci.gsfc.nasa.gov:9090/Fdata/INPUT/GVEG/ into *$WORKING/LDAS/input/GVEG/5km*

4. Download

   CLIM05_5KM.bin
   CLIM06_5KM.bin
   CLIM05_SAI_5KM.bin
   CLIM06_SAI_5KM.bin

   from http://lis2.sci.gsfc.nasa.gov:9090/Fdata/INPUT/AVHRR_LAI/ into *$WORKING/LDAS/input/AVHRR_LAI*

5. Change directory to *$WORKING/LDAS/input/GVEG/5km*. Unzip the UMD_605KM.txt.gz and UMD_60mask5KM.txt.gz files; i.e.,

```
% gzip -d UMD_605KM.txt.gz
% gzip -d UMD_60mask5KM.txt.gz
```

4.3 Scripts

The *scripts.tar.gz* file contains several scripts and parameter files used for running the LIS executable and for configuring the individual runs. These are described in Section 6.

Unpacking the *scripts.tar.gz* file will place the following files into the *$WORKING/LDAS* sub-directory:

File Name	Synopsis
BCS	Symbolic link to BCS data (input/BCS)
GVEG	Symbolic link to GVEG data (input/GVEG)
SRC	Symbolic link to src-par/driver
ldas.crd.clm.25	Sample card file
ldas.crd.clm.5km	Sample card file
ldas.crd.noah.25	Sample card file
ldas.crd.noah.5km	Sample card file
pbsinit.sh	Script to setup runs on an SGI Origin system
pcinit.sh	Script to setup runs on a Linux pc system
sgiinit.sh	Script to setup runs on an SGI workstation

4.4 Post-processing

The *postproc.tar.gz* file contains the source and data files needed to build and run the post-processing utility *mapto2D*. Post-processing is described in Section 7.

Unpacking the *postproc.tar.gz* file will create a *$WORKING/LDAS/postproc* sub-directory. The structure of *postproc* is as follows:

File Name	Synopsis
sgiinit.sh	Script to setup runs on an SGI workstation
compile.sh	Script to compile the mapto2D executable
gridout.ctl	Sample GrADS descriptor file
MASKS	Directory containing files used for 1d to 2d mapping
getmask.f90	
mapto2D.f90	
resolution_module.f90	
tile2grid.f90	

5 Building the Executable

This section describes how to build the source code and create LIS' executable, LIS.

First perform the steps described in Section 4.

If you are building on a Linux pc system, you must edit the *Makefile* located in *$WORKING/LDAS/src-par/make* Change the definition of `MPI_PREFIX` to the directory where you installed MPICH. Currently `MPI_PREFIX` is set to */data1/jim/local/mpich-1.2.4-absoft*.

Then

1. Change directory into *$WORKING/LDAS/src-par/make/MAKEDEP*
2. Run: `% gmake`
3. Change directory into *$WORKING/LDAS/src-par/make*
4. Run: `% gmake`
5. Move the resulting executable, named LIS, into the *$WORKING/LDAS* directory; i.e., `% mv LIS ../../`

See Appendix B to see the Makefile.

6 Running the Executable

This section describes how to run the LIS executable.

There are 4 different configurations that the code was run in for our Milestone F tests and performance runs. They are:

1. LIS running CLM at 1/4 deg – labeled clm.25
2. LIS running CLM at 5km – labeled clm.5km
3. LIS running NOAH at 1/4 deg – labeled noah.25
4. LIS running NOAH at 5km – labeled noah.5km

Each of these configurations performs a 1-day simulation (11 June 2001), computes with a time-step of 1800 seconds, uses GEOS forcing data, and generates output every 3 hours.

To run the code, first perform the steps described in Section 5 of this document.

6.1 SGI Origin

If you are running on an SGI Origin system at NASA Ames:

1. Change directory into *$WORKING/LDAS*
2. Run the PBS init shell script: `% sh pbsinit.sh user group`

   where user is your login name and group is your project's group id.
3. Run the appropriate shell script. For example, to run NOAH at 5km resolution, `% sh pbs.noah.5km.sh`

   These shell scripts will configure the run and submit the run into the queue on lomax.

Log files from these runs are written into the *$WORKING/LDAS* directory, and PBS output is written into the *$WORKING/LDAS* directory.

For example, running NOAH at 5km, using the *pbs.noah.5km.sh* shell script, will generate *$WORKING/LDAS/noah.5km.log*, *$WORKING/LDAS/pbs.noah.5km.out*, and *$WORKING/LDAS/pbs.noah.5km.err* log files.

6.2 SGI Workstation

If you are running on a local SGI workstation:

1. Change directory into *$WORKING/LDAS*
2. Run the SGI init shell script: `% sh sgiinit.sh`

3. Run the appropriate shell script. For example, to run NOAH at 5km resolution as a 32-process job, `% sh sgi.noah.5km.sh 32`

   These shell scripts will configure the run and launch the executable.

Log files from these runs are written into the *$WORKING/LDAS* directory.
For example, running NOAH at 5km, using the *sgi.noah.5km.sh* shell script, will generate a *$WORKING/LDAS/noah.5km.log* log file.

6.3 Linux PC

If you are running on a Linux PC:

1. Change directory into *$WORKING/LDAS*
2. Run the init shell script: `% sh pcinit.sh`
3. Run the appropriate shell script. For example, to run NOAH at 5km resolution as a 32-process job, `% sh pc.noah.5km.sh 32`

   These shell scripts will configure the run and launch the executable.

Log files from these runs are written into the *$WORKING/LDAS* directory.
For example, running NOAH at 5km, using the *pc.noah.5km.sh* shell script, will generate a *$WORKING/LDAS/noah.5km.log* log file.

6.4 Configuring Run Via LDAS Card File

This section describes how to configure your LIS run by manually editting the Fortran namelists contained in the *ldas.crd* "card file".

Currently, there are only a handful of options that may be reset using the card file. They are described in the following sub-sections. The remaining options should be left untouched.

If you manually edit the *ldas.crd* card file, do not run the LIS executable via the shell scripts described above. These scripts will over-write your newly editted *ldas.crd* card file. To run the LIS executable, read an appropriate shell script to see the necessary commands.

See Appendix A to see a sample ldas card file.

6.4.1 initial

There are no user-modifiable parameters in this namelist.

6.4.2 driver

In the driver namelist of the card file these parameters may be reset:

```
LDAS%d%DOMAIN
LDAS%m%LSM
LDAS%f%FORCE
```

`LDAS%d%DOMAIN` specifies the resolution for the run. Acceptable values are:

Value	Description
2	1/4 deg resolution
3	2×2.5 deg resolution
4	1 deg resolution
5	1/2 deg resolution
6	5 km resolution

`LDAS%m%LSM` specifies the land surface to run. Acceptable values are:

Value	Description
1	NOAH
2	CLM

`LDAS%f%FORCE` specifies the forcing data source for the run. Acceptable values are:

Value	Description
1	GDAS
2	GEOS

6.4.3 ldas_run_inputs

In the ldas_run_inputs namelist of the card file these parameters may be reset:

```
LDAS%o%EXPCODE
LDAS%t%SSS
LDAS%t%SMN
LDAS%t%SHR
LDAS%t%SDA
LDAS%t%SMO
LDAS%t%SYR
LDAS%t%ESS
LDAS%t%EMN
LDAS%t%EHR
LDAS%t%EDA
LDAS%t%EMO
LDAS%t%EYR
LDAS%t%TS
LDAS%o%ODIR
```

`LDAS%o%EXPCODE` specifies the "experiment code number" for the run. It is used in constructing the name of the output directory for the run. Acceptable values are any 3 digit integer string from 100 through 999. [2]

[2]Changing `LDAS%o%EXPCODE` from its default value of 999 will break the scripts contained in *scripts.tar.gz*.

Parameters LDAS%t%SSS, LDAS%t%SMN, LDAS%t%SHR, LDAS%t%SDA, LDAS%t%SMO, and LDAS%t%SYR are used in constructing the starting time for the run. Acceptable values are: [3]

Variable	Value	Description
LDAS%t%SSS	integer 0 – 59	specifying starting second
LDAS%t%SMN	integer 0 – 59	specifying starting minute
LDAS%t%SHR	integer 0 – 23	specifying starting hour
LDAS%t%SDA	integer 1 – 31	specifying starting day
LDAS%t%SMO	integer 1 – 12	specifying starting month
LDAS%t%SYR	integer 2001 – present	specifying starting year

Parameters LDAS%t%ESS, LDAS%t%EMN, LDAS%t%EHR, LDAS%t%EDA, LDAS%t%EMO, and LDAS%t%EYR are used in constructing the ending time for the run. Acceptable values are: [4]

Variable	Value	Description
LDAS%t%ESS	integer 0 – 59	specifying ending second
LDAS%t%EMN	integer 0 – 59	specifying ending minute
LDAS%t%EHR	integer 0 – 23	specifying ending hour
LDAS%t%EDA	integer 1 – 31	specifying ending day
LDAS%t%EMO	integer 1 – 12	specifying ending month
LDAS%t%EYR	integer 2001 – present	specifying ending year

LDAS%t%TS specifies the time-step for the run. Acceptable values are:

Value	Description
900	15 minute time-step
1800	30 minute time-step
3600	60 minute time-step

LDAS%o%ODIR specifies the name of the top-level output directory. Acceptable values are any 40 character string. [5] For simplicity, throughout the rest of this document, this top-level output directory shall be referred to by its default name, *$WORKING/LDAS/OUTPUT*.

6.4.4 gldas1_4

There are no user-modifiable parameters in this namelist.

[3]For this release of the code/data, the start time must be between 01 June 2001 and 30 June 2001

[4]For this release of the code/data, the end time must be between 01 June 2001 and 30 June 2001

[5]Changing LDAS%o%ODIR from its default value of OUTPUT will break the scripts contained in *scripts.tar.gz*.

6.4.5 gldas2x2_5

There are no user-modifiable parameters in this namelist.

6.4.6 gldas1

There are no user-modifiable parameters in this namelist.

6.4.7 gldas1_2

There are no user-modifiable parameters in this namelist.

6.4.8 nldas

There are no user-modifiable parameters in this namelist.

6.4.9 gldas5km

There are no user-modifiable parameters in this namelist.

6.4.10 clm2

There are no user-modifiable parameters in this namelist.

6.4.11 noah

In the noah namelist of the card file these parameters must correspond with the domain resolution set in the domain namelist:

```
LDAS%m%NOAH_MGFILE
LDAS%m%NOAH_ALBFILE
```

`LDAS%m%NOAH_MGFILE` and `LDAS%m%NOAH_ALBFILE` specify where to find certain NOAH related input parameter data files. Acceptable values are:

Value	Description
`"BCS/2x2.5deg/NOAH/"`	2×2.5 deg resolution
`"BCS/1deg/NOAH/"`	1 deg resolution
`"BCS/1_2deg/NOAH/"`	1/2 deg resolution
`"BCS/1_4deg/NOAH/"`	1/4 deg resolution
`"BCS/5km/NOAH/"`	5 km resolution

Both `LDAS%m%NOAH_MGFILE` and `LDAS%m%NOAH_ALBFILE` must have the same value.

6.4.12 geos

There are no user-modifiable parameters in this namelist.

6.4.13 gdas

There are no user-modifiable parameters in this namelist.

7 Output Data Processing

This section describes how to process the generated output.

The output datasets created by running the LIS executable are written into sub-directories of the *$WORKING/LDAS/OUTPUT/* directory (created at run-time). These sub-directories are named either *EXP999* (by default) or by the labels described in Section 6 (if run using the scripts described in Section 6).

The output data consists of ASCII text files and model output in binary format.

For example, assume that you performed a 1/4 deg NOAH run using the *sgi.noah.25.sh* shell script. This run will produce a *$WORKING/LDAS/OUTPUT/noah.25* directory. *noah.25* will contain:

File Name	Synopsis
ETAValidTime	Run-time diagnostic file for ETA valid time record (currently empty)
NOAHstats.dat	Statistical summary of output
fsource.dat	Forcing status summary (currently empty)
ldasdiag.dat	Run-time log file
NOAH	Directory containing output data

The *NOAH* directory will contain a *2001/20010611* sub-directory. Its contents are the output files generated by the executable. They are:

LDAS.E999.2001061100.NOAHgbin

LDAS.E999.2001061103.NOAHgbin

LDAS.E999.2001061106.NOAHgbin

LDAS.E999.2001061109.NOAHgbin

LDAS.E999.2001061112.NOAHgbin

LDAS.E999.2001061115.NOAHgbin

LDAS.E999.2001061118.NOAHgbin

LDAS.E999.2001061121.NOAHgbin

Note, each file-name contains a date-stamp marking the year, month, day, and hour that the data corresponds to. The output data files for CLM are similar.

The generated output is written in a binary format as a 1-d vector. It must be converted into a 2-d grid before it can be visualized, for example, with GrADS (see http://grads.iges.org/grads/).

7.1 Building mapto2D

This sub-section describes how to build the *mapto2D* executable used for converting the 1-d vector data into a 2-d grid.

To build the executable:

1. Change directory into *$WORKING/LDAS/postproc*
2. Run the compile.sh script: `% sh compile.sh`

7.2 Running mapto2D

This sub-section describes how to run the *mapto2D* executable used for converting the 1-d vector data into a 2-d grid.

To post-process the output files run:

1. Change directory into *$WORKING/LDAS/postproc*
2. Run *mapto2D*: `% mapto2D file res var`

   where:

   `file` specifies the name of the particular output file to process.

   `res` specifiles the resolution of the run. It must be the same value as `LDAS%d%DOMAIN`. See Section 6.4.2.

   `var` specifiles the number of the variable to extract and process. See Tables 7.3 and 7.4.

For example, assume that you performed a 1/4 deg NOAH run using the *sgi.noah.25.sh* shell script. To extract and process "Layer 1 Soil Temperature" at hour 03 of 11 June 2001:

```
% mapto2D ../OUTPUT/noah.25/NOAH/2001/20010611/LDAS.E999.2001061103.NOAHgbin
2 15
```

For this example, *mapto2D* will generate *gridoutV15.1gs4r*.

For those familiar with GrADS, there is a sample GrADS control file, *gridout.ctl*, included in the *$WORKING/LDAS/postproc* directory, which may be used to display this example.

7.3 CLM Output

Number	Variable	Units
1	Net Surface Shortwave Radiation	W/m^2
2	Net Surface Longwave Radiation	W/m^2
3	Latent Heat Flux	W/m^2
4	Sensible Heat Flux	W/m^2
5	Ground Heat Flux	W/m^2
6	Snow Phase Change Heat Flux	W/m^2
7	Downward Surface Shortwave Radiation	W/m^2
8	Downward Surface Longwave Radiation	W/m^2
9	Snowfall	kg/m^2
10	Rainfall	kg/m^2
11	Total Evaporation	kg/m^2
12	Surface Runoff	kg/m^2
13	Subsurface Runoff	kg/m^2
14	Snowmelt	kg/m^2
15	Snow Temperature	K
16	Canopy Temperature	K
17	Bare Soil Surface Temperature	K
18	Average Surface Temperature	K
19	Effective Radiative Surface Temperature	K
20	Surface Albedo, All Wavelengths	%
21	Snowpack Water Equivalent	kg/m^2
22	Plant Canopy Surface Water Storage	kg/m^2
23	Soil Temperature	K
24	Total Column Soil Moisture	kg/m^2
25	Root Zone Soil Moisture	kg/m^2
26	Top 1-meter Soil Moisture	kg/m^2
27	Soil Moisture	kg/m^2
28	Liquid Soil Moisture	kg/m^2
29	Total Soil Column Wetness	%
30	Root Zone Wetness	%

31	Canopy Surface Water Evaporation	W/m^2
32	Canopy Transpiration	W/m^2
33	Bare Soil Evaporation	W/m^2
34	Snow Evaporation	W/m^2
35	Potential Evaporation	W/m^2
36	Aerodynamic Conductance	m/s
37	Canopy Conductance	m/s
38	Leaf Area Index	unitless
39	Snow Depth	m
40	Snow Cover	%
41	Snow Albedo	%
42	Two Meter Temperature	K
43	Two Meter Humidity	kg/kg
44	Ten Meter U Wind	m/s
45	Ten Meter V Wind	m/s
46	Surface Pressure	mb

7.4 NOAH Output

Number	Variable	Units
1	Two Meter Air Temperature	K
2	Two Meter Specific Humidity	kg/kg
3	Surface Pressure	Pa
4	Snowfall (frozen precipitation)	kg/m^2
5	Rainfall (unfrozen precipitation)	kg/m^2
6	Downward Surface Shortwave Radiation	W/m^2
7	Downward Surface Longwave Radiation	W/m^2
8	Net shortwave radiation (surface)	W/m^2
9	Net longwave radiation (surface)	W/m^2
10	Ten Meter U Wind	m/s
11	Ten Meter V Wind	m/s
12	Convective Precipitation	kg/m^2
13	Canopy water Content	kg/m^2
14	Skin Temperature	K
15	Layer 1 Soil Temperature	K
16	Layer 2 Soil Temperature	K
17	Layer 3 Soil Temperature	K
18	Layer 4 Soil Temperature	K
19	Layer 1 Vol. Soil Moisture = liq+frzn	kg/m^2
20	Layer 2 Vol. Soil Moisture = liq+frzn	kg/m^2
21	Layer 3 Vol. Soil Moisture = liq+frzn	kg/m^2
22	Layer 4 Vol. Soil Moisture = liq+frzn	kg/m^2
23	Layer 1 Vol. Liquid Soil Moisture	kg/m^2
24	Layer 2 Vol. Liquid Soil Moisture	kg/m^2
25	Layer 3 Vol. Liquid Soil Moisture	kg/m^2
26	Layer 4 Vol. Liquid Soil Moisture	kg/m^2
27	Snow Depth	m
28	Water Equivalent Snow Depth	kg/m^2

29	Surface Exchange Coef. for Heat and Moisture	unitless
30	Surface Exchange Coef. for Momentum	unitless
31	Surface Albedo fraction	%
32	Vegetation Greenness	%
33	Latent Heat Flux	W/m^2
34	Sensible Heat Flux	W/m^2
35	Canopy evaporation	W/m^2
36	Direct Soil Evaporation	W/m^2
37	Transpiration	W/m^2
38	Sublimation from Snowpack	W/m^2
39	Total Evaporation	kg/m^2
40	Excess Canopy Moisture	m
41	Dewfall Amount	m s^{-1}
42	Ratio of actual/potential EVAP	unitless
43	Final Potential Evapotranspiration	W/m^2
44	Ground heat flux	W/m^2
45	Snow phase-change heatflux	W/m^2
46	Snow melt (water equivalent)	kg/m^2
47	Fractional Snow Cover	Unitless fraction
48	Storm surface runoff	kg/m^2
49	Baseflow-groundwater runoff	kg/m^2
50	Canopy resistence	s m^{-1}
51	Canopy conductance	m s^{-1}
52	Avail Soil Moist in Root Zone	%
53	Avail Soil Moist in Total Col	%
54	Total Col Soil Moist Content	kg/m^2
55	Root Zone Col Soil Moist Content	kg/m^2
56	Top 1-Meter Col Soil Moist Content	kg/m^2

8 Retrieving "F" Output Data

This section describes how to download datasets generated by LIS' development team for LIS' "First Code Improvement" milestone – Milestone F.

Two output datasets are available for download, one from a 5 km NOAH run and another from a 5 km CLM run. The 5 km NOAH datasets are located at http://lis2.sci.gsfc.nasa.gov:9090/Fdata/OUTPUT/NOAH-5km/, and the 5 km CLM datasets are located at http://lis2.sci.gsfc.nasa.gov:9090/Fdata/OUTPUT/CLM-5km/.

The output data files for NOAH are:

LDAS.E999.2001061100.NOAHgbin

LDAS.E999.2001061103.NOAHgbin

LDAS.E999.2001061106.NOAHgbin

LDAS.E999.2001061109.NOAHgbin

LDAS.E999.2001061112.NOAHgbin

LDAS.E999.2001061115.NOAHgbin

LDAS.E999.2001061118.NOAHgbin

LDAS.E999.2001061121.NOAHgbin

Note, each file-name contains a date-stamp marking the year, month, day, and hour that the data corresponds to. The output data files for CLM are similar.

These output data files are large and require post-processing before reading them, see Section 7. Therefore several data plots have been generated, and they are found on LIS' web-site at http://lis.gsfc.nasa.gov/Documentation/Documents/output/.

A LDAS Card File

```
&initial
LDAS%r%RCLM2       = 0
LDAS%r%RNOAH       = 0
LDAS%f%FGEOS       = 0
LDAS%f%FGDAS       = 0
LDAS%p%KOSTER      = 0
LDAS%p%PRECSOR     = 0,99,99,99
/

&driver
LDAS%d%DOMAIN = 3
LDAS%m%LSM    = 2
LDAS%f%FORCE  = 2
LDAS%r%SOIL   = 2
LDAS%p%LAI    = 2
/

&ldas_run_inputs
LDAS%o%EXPCODE    = 999
LDAS%p%NT         = 13
LDAS%f%NF         = 10
LDAS%f%NMIF       = 15
LDAS%o%WFOR       = 0
LDAS%o%WTIL       = 0
LDAS%o%WHDF       = 0
LDAS%o%WGRB       = 0
LDAS%o%WBIN       = 1
LDAS%o%STARTCODE  = 3
LDAS%t%SSS        = 0
LDAS%t%SMN        = 00
LDAS%t%SHR        = 21
LDAS%t%SDA        = 10
LDAS%t%SMO        = 06
LDAS%t%SYR        = 2001
LDAS%t%ENDCODE    = 1
LDAS%t%ESS        = 0
LDAS%t%EMN        = 00
LDAS%t%EHR        = 21
LDAS%t%EDA        = 11
LDAS%t%EMO        = 06
LDAS%t%EYR        = 2001
LDAS%t%TS         = 1800
LDAS%d%UDEF       = -9999.
LDAS%o%WRITEINTF  = 3.
```

```
LDAS%o%ODIR        = "OUTPUT"
LDAS%o%DFILE       = "ldasdiag.dat"
LDAS%o%FFILE       = "fsource.dat"
LDAS%o%EVTFILE     = "ETAValidTime"
LDAS%p%VCLASS      = 1
LDAS%p%AVHRR       = "./input/AVHRR_LAI"
/

&gldas1_4
LDAS%d%NC          = 1440
LDAS%d%NR          = 600
LDAS%d%MAXT        = 1
LDAS%d%MINA        = 0.05
LDAS%p%VFILE       = "GVEG/1_4deg/UMD_AVHRR60G0.25.txt"
LDAS%p%MFILE       = "GVEG/1_4deg/UMD_AVHRR60mask0.25.asc"
LDAS%p%FMFILE      = "GVEG/1_4deg/UMD_AVHRR60mask0.25.asc"
LDAS%p%SFILE       = "BCS/1_4deg/sim60soil0.25.txt"
LDAS%p%SAFILE      = "BCS/1_4deg/sand60.25.bfsa"
LDAS%p%CLFILE      = "BCS/1_4deg/clay60.25.bfsa"
LDAS%p%POFILE      = "BCS/1_4deg/por60.25.bfsa"
LDAS%p%ISCFILE     = "BCS/1_4deg/soicol60.25i.bin"
LDAS%p%KVFILE      = "GVEG/1_4deg/tile_info.1440x600"
LDAS%p%KOSTER      = 0
/

&gldas2x2_5
LDAS%d%NC          = 144
LDAS%d%NR          = 76
LDAS%d%MAXT        = 1
LDAS%d%MINA        = 0.05
LDAS%p%VFILE       = "GVEG/2x2.5deg/UMD_AVHRR60G2-2.5.txt"
LDAS%p%MFILE       = "GVEG/2x2.5deg/UMD_AVHRR60mask2-2.5.asc"
LDAS%p%FMFILE      = "GVEG/2x2.5deg/UMD_AVHRR60mask2-2.5.asc"
LDAS%p%SFILE       = "BCS/2x2.5deg/sim60soil2-2.5.txt"
LDAS%p%SAFILE      = "BCS/2x2.5deg/sand60_2x2.5.bfsa"
LDAS%p%CLFILE      = "BCS/2x2.5deg/clay60_2x2.5.bfsa"
LDAS%p%POFILE      = "BCS/2x2.5deg/por60_2x2.5.bfsa"
LDAS%p%ISCFILE     = "BCS/2x2.5deg/soicol60_2x2.5i.bin"
LDAS%p%KVFILE      = "GVEG/2x2.5deg/tile_info.144x76"
LDAS%p%KOSTER      = 0
/

&gldas1
LDAS%d%NC          = 360
LDAS%d%NR          = 150
LDAS%d%MAXT        = 1
```

```
LDAS%d%MINA        = 0.05
LDAS%p%VFILE       = "GVEG/1deg/UMD_AVHRR60G1.0.txt"
LDAS%p%MFILE       = "GVEG/1deg/UMD_AVHRR60mask1.0.asc"
LDAS%p%FMFILE      = "GVEG/1deg/UMD_AVHRR60mask1.0.asc"
LDAS%p%SFILE       = "BCS/1deg/sim60soil1.0.txt"
LDAS%p%SAFILE      = "BCS/1deg/sand60_1.0.bfsa"
LDAS%p%CLFILE      = "BCS/1deg/clay60_1.0.bfsa"
LDAS%p%POFILE      = "BCS/1deg/por60_1.0.bfsa"
LDAS%p%ISCFILE     = "BCS/1deg/soicol60_1.0i.bin"
LDAS%p%KVFILE      = "GVEG/1deg/tile_info.144x76"
LDAS%p%KOSTER      = 0
/

&gldas1_2
LDAS%d%NC          = 720
LDAS%d%NR          = 300
LDAS%d%MAXT        = 1
LDAS%d%MINA        = 0.05
LDAS%p%VFILE       = "GVEG/1_2deg/UMD_AVHRR60G0.50.txt"
LDAS%p%MFILE       = "GVEG/1_2deg/UMD_AVHRR60mask0.50.asc"
LDAS%p%FMFILE      = "GVEG/1_2deg/UMD_AVHRR60mask0.50.asc"
LDAS%p%SFILE       = "BCS/1_2deg/sim60soil0.5.txt"
LDAS%p%SAFILE      = "BCS/1_2deg/sand60_0.5.bfsa"
LDAS%p%CLFILE      = "BCS/1_2deg/clay60_0.5.bfsa"
LDAS%p%POFILE      = "BCS/1_2deg/por60_0.5.bfsa"
LDAS%p%ISCFILE     = "BCS/1_2deg/soicol60_0.5i.bin"
LDAS%p%KVFILE      = "GVEG/1_2deg/tile_info.144x76"
LDAS%p%KOSTER      = 0
/

&nldas
LDAS%d%NC          = 464
LDAS%d%NR          = 224
LDAS%d%MAXT        = 1
LDAS%d%MINA        = 0.05
LDAS%p%VFILE       = "BCS/UMD_LDAS.txt"
LDAS%p%MFILE       = "BCS/Mask.UMD_LDAS.txt"
LDAS%p%FMFILE      = "BCS/us.map.txt"
LDAS%p%SFILE       = "BCS/LDAS_SOILS.txt"
LDAS%p%SAFILE      = "BCS/sand_nldas2m.bfsa"
LDAS%p%CLFILE      = "BCS/clay_nldas2m.bfsa"
LDAS%p%POFILE      = "BCS/por_nldas2m.bfsa"
LDAS%p%ISCFILE     = "BCS/soicol_nldasi.bin"
LDAS%f%MERGEH      = "/LDAS/DATA/MERGEH"
LDAS%p%PRECSOR     = 0,99,99,99
/
```

```
&gldas5km
LDAS%d%NC          = 7200
LDAS%d%NR          = 3000
LDAS%d%MAXT        = 1
LDAS%d%MINA        = 0.05
LDAS%p%VFILE       = "GVEG/5km/UMD_605KM.txt"
LDAS%p%MFILE       = "GVEG/5km/UMD_60mask5KM.txt"
LDAS%p%FMFILE      = "GVEG/5km/UMD_60mask5KM.txt"
LDAS%p%SFILE       = "BCS/5km/sim60soil5KM.asc"
LDAS%p%SAFILE      = "BCS/5km/sand605KM.bfsa"
LDAS%p%CLFILE      = "BCS/5km/clay605KM.bfsa"
LDAS%p%POFILE      = "BCS/5km/por605KM.bfsa"
LDAS%p%ISCFILE     = "BCS/5km/soicol605KM.bin"
LDAS%p%KVFILE      = "GVEG/5km/tile_info.1440x600"
LDAS%p%KOSTER      = 0
/

&clm2
LDAS%r%RCLM2         = 1
LDAS%o%WRITEINTC2    = 3
LDAS%m%CLM2_RFILE    = "clm2.rst"
LDAS%m%CLM2_VFILE    = "BCS/clm_parms/umdvegparam.txt"
LDAS%m%CLM2_CHTFILE = "BCS/clm_parms/clm2_ptcanhts.txt"
LDAS%m%CLM2_OFILE    = "clm2_out.dat"
LDAS%m%CLM2_PFILE    = "CLM2_POUT.DAT"
LDAS%m%CLM2_ISM      = 0.45
LDAS%m%CLM2_IT       = 290.0
LDAS%m%CLM2_ISCV     = 0.
/

&noah
LDAS%r%RNOAH         = 1
LDAS%o%WRITEINTN     = 3
LDAS%m%NOAH_RFILE    = "noah.rst"
LDAS%m%NOAH_MFILE    = "noahgdas.rst"
LDAS%m%NOAH_MGFILE  = "BCS/1_4deg/NOAH/"
LDAS%m%NOAH_ALBFILE = "BCS/1_4deg/NOAH/"
LDAS%m%NOAH_VFILE    = "BCS/noah_parms/noah.vegparms.txt"
LDAS%m%NOAH_SFILE    = "BCS/noah_parms/noah.soilparms.txt"
LDAS%m%NOAH_NMXSNAL = "BCS/1_8deg/NOAH/maxsnalb_0.125.bin"
LDAS%m%NOAH_GMXSNAL = "BCS/1_4deg/NOAH/maxsnalb_0.25.bfsa"
LDAS%m%NOAH_SMXSNAL = "BCS/2x2.5deg/NOAH/maxsnalb_2-2.5.bfsa"
LDAS%m%NOAH_OMXSNAL = "BCS/1deg/NOAH/maxsnalb_1.0.bfsa"
LDAS%m%NOAH_EMXSNAL = "BCS/1_2deg/NOAH/maxsnalb_0.5.bfsa"
LDAS%m%NOAH_YMXSNAL = "BCS/5km/NOAH/maxsnalb_5KM.bfsa"
```

```
LDAS%m%NOAH_NTBOT   = "BCS/1_8deg/NOAH/tbot_0.125.bin"
LDAS%m%NOAH_GTBOT   = "BCS/1_4deg/NOAH/tbot_0.25.bfsa"
LDAS%m%NOAH_STBOT   = "BCS/2x2.5deg/NOAH/tbot_2-2.5.bfsa"
LDAS%m%NOAH_OTBOT   = "BCS/1deg/NOAH/tbot_1.0.bfsa"
LDAS%m%NOAH_ETBOT   = "BCS/1_2deg/NOAH/tbot_0.5.bfsa"
LDAS%m%NOAH_YTBOT   = "BCS/5km/NOAH/tbot-uncr_5KM.bfsa"
LDAS%m%NOAH_ISM     = 0.30
LDAS%m%NOAH_IT      = 290.0
LDAS%m%NOAH_IC      = 3
LDAS%m%NOAH_SMDA    = 0
LDAS%m%NOAH_SDA     = 0
LDAS%m%NOAH_TDA     = 0
LDAS%m%NOAH_NVEGP   = 7
LDAS%m%NOAH_NMVEGP  = 1
LDAS%m%NOAH_NSOILP  = 10
LDAS%m%NOAH_ZST     = 9
LDAS%m%NOAH_NRET    = 56
/

&geos
LDAS%f%FGEOS      = 1
LDAS%f%GEOSDIR    = "./input/FORCING/GEOS/BEST_LK"
LDAS%f%GEOSTIME1  = 3000.0
LDAS%f%GEOSTIME2  = 0.0
LDAS%f%NROLD      = 181
LDAS%f%NCOLD      = 360
LDAS%f%NMIF       = 13
/

&gdas
LDAS%f%FGDAS      = 1
LDAS%f%GDASDIR    = "./input/FORCING/GDAS"
LDAS%f%GDASTIME1  = 3000.0
LDAS%f%GDASTIME2  = 0.0
LDAS%f%NROLD      = 256
LDAS%f%NCOLD      = 512
LDAS%f%NMIF       = 15
/
```

B Makefile

```
# Set up special characters

null  :=
space := $(null) $(null)

# Check for directory in which to put executable
ifeq ($(MODEL_EXEDIR),$(null))
MODEL_EXEDIR := .
endif

# Check for name of executable
ifeq ($(EXENAME),$(null))
EXENAME := LIS
endif

# Check if SPMD is defined in "misc.h"
# Ensure that it is defined and not just "undef SPMD" set in file
ifeq ($(SPMD),$(null))
  SPMDSET := $(shell /bin/grep SPMD misc.h)
  ifneq (,$(findstring define,$(SPMDSET)))
    SPMD := TRUE
  else
    SPMD := FALSE
  endif
endif

# Determine platform
UNAMES := $(shell uname -s)
UMACHINE := $(shell uname -m)

ifeq ($(UNAMES),IRIX64)

#LIB_NETCDF := /u/jvg/local/netcdf-3.5.0/lib
#INC_NETCDF := /u/jvg/local/netcdf-3.5.0/include
#LIB_NETCDF := /ford1/local/irix6.2/netcdf/lib
#INC_NETCDF := /ford1/local/irix6.2/netcdf/include
#LIB_MPI    := /ford1/local/irix6.2/mpi/lib/IRIX64/ch_p4
#INC_MPI    := /ford1/local/irix6.2/mpi/include/IRIX64/ch_p4
ESMF_DIR   := ../lib/esmf
LIB_ESMF   := $(ESMF_DIR)/lib/libO
MOD_ESMF   := $(ESMF_DIR)/mod/modO

endif
```

```
ifeq ($(UNAMES),OSF1)

LIB_NETCDF := /usr/ulocal/netcdf-3.5.0/lib
INC_NETCDF := /usr/ulocal/netcdf-3.5.0/include
LIB_MPI := /usr/lib
INC_MPI := /usr/include
ESMF_DIR   := /scr/sujay/LIS/LDAS/SRC/esmf
LIB_ESMF := $(ESMF_DIR)/lib/libg
MOD_ESMF := $(ESMF_DIR)/mod/modg

endif

ifeq ($(UMACHINE), i686)

#LIB_NETCDF := /usr/ulocal/netcdf-3.5.0/lib
#INC_NETCDF := /usr/ulocal/netcdf-3.5.0/include
MPI_PREFIX := /data1/jim/local/mpich-1.2.4-absoft
LIB_MPI := $(MPI_PREFIX)/lib
INC_MPI := $(MPI_PREFIX)/include
ESMF_DIR := ../lib/esmf
LIB_ESMF := $(ESMF_DIR)/lib/libO
MOD_ESMF := $(ESMF_DIR)/mod/modO

endif

# Load dependency search path.
dirs := . $(shell cat Filepath)
# Set cpp search path, include netcdf
cpp_dirs := $(dirs) $(INC_NETCDF) $(INC_MPI)
cpp_path := $(foreach dir,$(cpp_dirs),-I$(dir)) # format for command line

# Expand any tildes in directory names. Change spaces to colons.
VPATH    := $(foreach dir,$(cpp_dirs),$(wildcard $(dir)))
VPATH    := $(subst $(space),:,$(VPATH))

#-----------------------------------------------------------------------
# Primary target: build the model
#-----------------------------------------------------------------------
all: $(MODEL_EXEDIR)/$(EXENAME)

# Get list of files and determine objects and dependency files
FIND_FILES = $(wildcard $(dir)/*.F $(dir)/*.f $(dir)/*.F90 $(dir)/*.c)
FILES       = $(foreach dir, $(dirs),$(FIND_FILES))
SOURCES    := $(sort $(notdir $(FILES)))
DEPS       := $(addsuffix .d, $(basename $(SOURCES)))
OBJS       := $(addsuffix .o, $(basename $(SOURCES)))
```

```
$(MODEL_EXEDIR)/$(EXENAME): $(OBJS)
 $(FC) -o $@ $(OBJS) $(FOPTS) $(LDFLAGS)
debug: $(OBJS)
        echo "FFLAGS: $(FFLAGS)"
        echo "LDFLAGS: $(LDFLAGS)"
        echo "OBJS: $(OBJS)"

#***********************************************************************
#********** Architecture-specific flags and rules**********************
#***********************************************************************

#------------------------------------------------------------------------
# SGI
#------------------------------------------------------------------------


ifeq ($(UNAMES),IRIX64)

ESMF_ARCH = IRIX64
FC          := f90
CPP         := /lib/cpp

# Library directories
LIB_DIR  = ../lib/sgi-64/
HDFLIBDIR  = $(LIB_DIR)
GFIOLIBDIR = $(LIB_DIR)
CPPFLAGS    := -P
PSASINC     :=
CFLAGS      := $(cpp_path) -DIRIX64 -O2 -OPT:Olimit=0 -static
#CFLAGS      := $(cpp_path) -DIRIX64 -g -static
FFLAGS      = $(cpp_path) -64 -r4 -i4 -c -cpp -I$(MOD_ESMF)/$(ESMF_ARCH) -DHIDE_SHR_MSG -DNO
#FFLAGS      = $(cpp_path) -64 -r4 -i4 -c -cpp -I$(MOD_ESMF)/$(ESMF_ARCH) -DHIDE_SHR_MSG -DN
#FFLAGS      = $(cpp_path) -64 -r4 -i4 -c -cpp -I$(MOD_ESMF)/$(ESMF_ARCH) -DHIDE_SHR_MSG -DN
FOPTS = $(LIB_DIR)bacio_64_sgi $(LIB_DIR)w3lib_64_sgi
#LDFLAGS     = -L$(LIB_NETCDF) -lnetcdf -L$(LIB_ESMF)/$(ESMF_ARCH) -loldworld -lmpi
#LDFLAGS     = -L$(LIB_ESMF)/$(ESMF_ARCH) -loldworld -lmpi
LDFLAGS     = -64 -L$(LIB_ESMF)/$(ESMF_ARCH) -lesmf -lmpi
# WARNING: -mp and -g together cause wrong answers

# WARNING: - Don’t run hybrid on SGI (that’s what the -= -mp is all about)

ifeq ($(SPMD),TRUE)
  FFLAGS  -= -mp
  FFLAGS  += -macro_expand
#  FFLAGS  += -I$(INC_MPI) -macro_expand
```

```
#  LDFLAGS += -L$(LIB_MPI) -lmpi
else
  FFLAGS  += -DHIDE_MPI
endif

.SUFFIXES:
.SUFFIXES: .F .F90 .f .f90 .c .o

.F.o:
$(FC) $(FFLAGS) $<
.F90.o:
$(FC) $(FFLAGS) $<
.f.o:
$(FC) $(FFLAGS) $<
.f90.o:
$(FC) $(FFLAGS) $<
.c.o:
cc -64 -c $(cpp_path) $(CFLAGS) $<

endif

#---------------------------------------------------------------------------
# Linux
#---------------------------------------------------------------------------


ifeq ($(UMACHINE),i686)

ESMF_ARCH = linux_absoft

ifeq ($(ESMF_ARCH),linux_ifc)

FC          := $(MPI_PREFIX)/bin/mpif90
CPP         := /lib/cpp

CFLAGS      := $(cpp_path) -O2
FFLAGS      = $(cpp_path) -c -I$(MOD_ESMF)/$(ESMF_ARCH) -DHIDE_SHR_MSG -DNO_SHR_VMATH -O
LDFLAGS     = -L$(LIB_ESMF)/$(ESMF_ARCH) -lesmf -lmpich

endif

ifeq ($(ESMF_ARCH),linux_absoft)

FC          := $(MPI_PREFIX)/bin/mpif90
CPP         := /lib/cpp
```

```
CFLAGS      := $(cpp_path) -O2 -DABSOFT
FFLAGS      = $(cpp_path) -c -O2 -YEXT_NAMES=LCS -s -B108 -YCFRL=1 -p$(MOD_ESMF)/$(ESMF_ARCH
#FFLAGS      = $(cpp_path) -c -O2 -YEXT_NAMES=LCS -s -B108 -YCFRL=1 -p$(MOD_ESMF)/$(ESMF_ARC
#FFLAGS      = $(cpp_path) -c -O1 -g -Rb -Rc -Rs -Rp -YEXT_NAMES=LCS -s -B108 -YCFRL=1 -p$(M
#CFLAGS      := $(cpp_path) -g -DABSOFT
#FFLAGS      = $(cpp_path) -c -g -YEXT_NAMES=LCS -s -B108 -YCFRL=1 -p$(MOD_ESMF)/$(ESMF_ARCH
LDFLAGS     = -L$(LIB_ESMF)/$(ESMF_ARCH) -lesmf -lmpich -lU77

endif

# Library directories
LIB_DIR  = ../lib/pc-32/
HDFLIBDIR  = $(LIB_DIR)
GFIOLIBDIR = $(LIB_DIR)
CPPFLAGS    := -P
PSASINC     :=
FOPTS = $(LIB_DIR)bacio_32_pclinux $(LIB_DIR)w3lib_32_pclinux
# WARNING: -mp and -g together cause wrong answers

# WARNING: - Don't run hybrid on SGI (that's what the -= -mp is all about)

ifeq ($(SPMD),TRUE)
#  FFLAGS  -= -mp
#  FFLAGS  += -macro_expand
#  FFLAGS  += -I$(INC_MPI) -macro_expand

#  LDFLAGS += -L$(LIB_MPI) -lmpi
else
  FFLAGS  += -DHIDE_MPI
endif

.SUFFIXES:
.SUFFIXES: .F .F90 .f .f90 .c .o

.F.o:
$(FC) $(FFLAGS) $<
.F90.o:
$(FC) $(FFLAGS) $<
.f.o:
$(FC) $(FFLAGS) $<
.f90.o:
$(FC) $(FFLAGS) $<
.c.o:
cc -c $(cpp_path) $(CFLAGS) $<
```

```
endif

#------------------------------------------------------------------------
# Targets/rules that depend on architecture specific variables.
#------------------------------------------------------------------------

# The library is not made to be built in parallel, so we must null passed options and
# specify only one job is to run.
$(LIB_ESMF)/$(ESMF_ARCH)/libesmf.a:
        cd $(ESMF_SRCDIR); \
           $(MAKE) -j 1 BOPT=$(BOPT_ESMF) MF_BUILD=$(ESMF_LIBDIR) MF_DIR=$(ESMF_SRCDIR)
MF_ARCH=$(ESMF_ARCH);

time_manager.o : $(LIB_ESMF)/$(ESMF_ARCH)/libesmf.a

RM := rm
# Add user defined compiler flags if set, and replace FC if USER option set.
FFLAGS  += $(USER_FFLAGS)
ifneq ($(USER_FC),$(null))
FC := $(USER_FC)
endif

clean:
$(RM) -f *.o *.mod *.stb *.f90 $(MODEL_EXEDIR)/$(EXENAME)

realclean:
$(RM) -f *.o *.d *.mod *.stb *.f90 $(MODEL_EXEDIR)/$(EXENAME)


#------------------------------------------------------------------------
#!!!!!!!!!!!!!!!!DO NOT EDIT BELOW THIS LINE.!!!!!!!!!!!!!!!!!!!!!!!!!!!!
#------------------------------------------------------------------------
# These rules cause a dependency file to be generated for each source
# file.  It is assumed that the tool "makdep" (provided with this
# distribution in clm2/tools/makdep) has been built and is available in
# the user's $PATH.  Files contained in the clm2 distribution are the
# only files which are considered in generating each dependency.  The
# following filters are applied to exclude any files which are not in
# the distribution (e.g. system header files like stdio.h).
#
#  1) Remove full paths from dependencies. This means gnumake will not break
#     if new versions of files are created in the directory hierarchy
#     specified by VPATH.
#
#  2) Because of 1) above, remove any file dependencies for files not in the
#     clm2 source distribution.
```

```
#
# Finally, add the dependency file as a target of the dependency rules.  This
# is done so that the dependency file will automatically be regenerated
# when necessary.
#
#     i.e. change rule
#       make.o : make.c make.h
#       to:
#       make.o make.d : make.c make.h
#------------------------------------------------------------------------
DEPGEN := ./MAKDEP/makdep -s F
%.d : %.c
@echo "Building dependency file $@"
@$(DEPGEN) -f $(cpp_path) $< > $@
%.d : %.f
@echo "Building dependency file $@"
@$(DEPGEN) -f $(cpp_path) $< > $@
%.d : %.F90
@echo "Building dependency file $@"
@$(DEPGEN) -f $(cpp_path) $<  > $@
%.d : %.F
@echo "Building dependency file $@"
@$(DEPGEN) -f $(cpp_path) $< > $@
#
# if goal is clean or realclean then don't include .d files
# without this is a hack, missing dependency files will be created
# and then deleted as part of the cleaning process
#
INCLUDE_DEPS=TRUE
ifeq ($(MAKECMDGOALS), realclean)
 INCLUDE_DEPS=FALSE
endif
ifeq ($(MAKECMDGOALS), clean)
 INCLUDE_DEPS=FALSE
endif

ifeq ($(INCLUDE_DEPS), TRUE)
-include $(DEPS)
endif
```

References

[1] F. Chen, K. Mitchell, J. Schaake, Y. Xue, H. Pan, V. Koren, Y. Duan, M. Ek, and A. Betts. Modeling of land-surface evaporation by four schemes and comparison with fife observations. *J. Geophys. Res.*, 101(D3):7251–7268, 1996.

[2] G. J. Collatz, C. Grivet, J. T. Ball, and J. A. Berry. Physiological and environmental regulation of stomatal conducatance: Photosynthesis and transpiration: A model that includes a laminar boundary layer. *Agric. For. Meteorol.*, 5:107–136, 1991.

[3] P. G. Jarvis. The interpretation of leaf water potential and stomatal conductance found in canopies of the field. *Phil. Trans. R. Soc.*, B(273):593–610, 1976.

[4] L. A. Richards. Capillary conduction of liquids in porous media. *Physics*, 1:318–333, 1931.

[5] E. Rogers, T. L. Black, D. G. Deaven, G. J. DiMego, Q. Zhao, M. Baldwin, N. W. Junker, and Y. Lin. Changes to the operational "early" eta analysis/forecast system at the national centers of environmental prediction. *Wea. Forecasting*, 11:391–413, 1996.